Automata and Formal Languages. Homework 10 (18.11.2024)

1. Describe (informally) a semi-algorithm either for the positive or for the negative instances of the following problems:

   (a) Are given context-free languages L_1 and L_2 equal ?

   (b) Do given context-free languages L_1 and L_2 contain at least one common word ?

   (c) Is a given context-free grammar G ambiguous ?

   (d) Does a given Turing machine M have a periodic ID. (The ID α is called periodic if $\alpha \vdash^+ \alpha$.)

2. Show all the moves of the Turing machine $M = (\{s, q, p, f\}, \{a\}, \{a, B\}, \delta, s, B, f)$ from the initial configuration sB until it halts, where

$$\begin{aligned} \delta(s, B) &= (q, a, R) \\ \delta(s, a) &= (f, a, R) \\ \delta(q, B) &= (q, a, L) \\ \delta(q, a) &= (p, B, R) \\ \delta(p, B) &= (p, a, L) \\ \delta(p, a) &= (s, a, L) \end{aligned}$$

   (Don't give up! But if you have done more than 30 moves you made a mistake.) Show all the intermediate ID's. How many steps does the machine make before halting ? What is the language the machine recognizes ?

3. Construct a Turing machine that recognizes the language

$$L = \{a^{(2^n)} \mid n \geq 0\}.$$

   Give the full transition function δ, and give the accepting computation (all ID's) for input word aa.

4. Construct a TM with the tape alphabet $\{B, \$\}$ that "finds the money": the machine is started in its initial state q_0 anywhere on the tape. If the tape is completely blank the machine never halts. If there is money \$ somewhere on the tape, the machine halts in its final state.

   Note: In the beginning, the money may be arbitrarily far to the right or to the left of the machine. You may not introduce new tape symbols.

5. Consider the following function f from positive integers to positive integers:

$$f(n) = \begin{cases} n/2, & \text{if } n \text{ is even,} \\ 3n + 1, & \text{if } n \text{ is odd.} \end{cases}$$

   Let us iterate the function, starting with some n, until 1 is reached, if ever. For example, starting with $n = 3$ we get the following sequence:

$$3 \longrightarrow 10 \longrightarrow 5 \longrightarrow 16 \longrightarrow 8 \longrightarrow 4 \longrightarrow 2 \longrightarrow 1.$$

   It is a famous open problem (known as the Collatz problem) whether every positive integer eventually iterates to 1, that is, whether

$$L = \{a^n \mid \exists i : f^i(n) = 1\}$$

   is the language a^+. In this problem we design a Turing machine that recognizes the language L. To help the task, the TM program is divided into four small "subroutines".

   Therefore, construct (in full details) Turing machines with the following behaviors. In each case, any number of new states and intermediate tape symbols may be used. The machine may act in arbitrary fashion on ID's that do not have the specified format:

(a) Initialize computation by adding markers at both ends of the input:

$$q_0 a^n \vdash^* \#q_1 a^n\$$$

for every $n \geq 1$,

(b) Evaluate the parity of the number, and accept if $n = 1$:

$$\#q_1 a^n\$ \vdash^* \begin{cases} \#fa\$, & \text{if } n = 1, \\ \#q_{even} a^n\$, & \text{if } n > 1 \text{ is even}, \\ \#q_{odd} a^n\$, & \text{if } n > 1 \text{ is odd}. \end{cases}$$

(c) If $n > 1$ is even divide it by 2:

$$\#q_{even} a^{2n}\$ \vdash^* \#q_1 a^n\$$$

(d) If $n > 1$ is odd multiply by three and add one:

$$\#q_{odd} a^n\$ \vdash^* \#q_1 a^{3n+1}\$$$

6. Prove that for every recursively enumerable language L there exists

   (a) a recursive language N and a homomorphism h such that $L = h(N)$.

   (b) a recursive language K and a regular language R such that $L = K/R$.

   To prove that a language N or K is recursive it is sufficient to describe an algorithm to determine if a given word is in the language – no need to construct a TM for the language. (Hint: append to words $w \in L$ suffixes that indicate the number of moves of the TM on input w.)

7. The tape of a *two-dimensional Turing machine* is an infinite square grid, like an infinite graph paper. The squares (=cells) are indexed by $\mathbb{Z}^2$. The Turing machine moves on the grid, and based on the current state and the symbol written on the current cell the machine changes its state, writes a new symbol into the cell and moves to one of the four neighboring cells, as indicated by the transition function.

   Langton's ant is the following two-dimensional TM: The state set is $Q = \{\leftarrow, \uparrow, \rightarrow, \downarrow\}$ and the tape alphabet is $\Gamma = \{\text{Black}, \text{White}\}$. The arrow gives the current orientation of the ant. The transition function is described by the following steps:

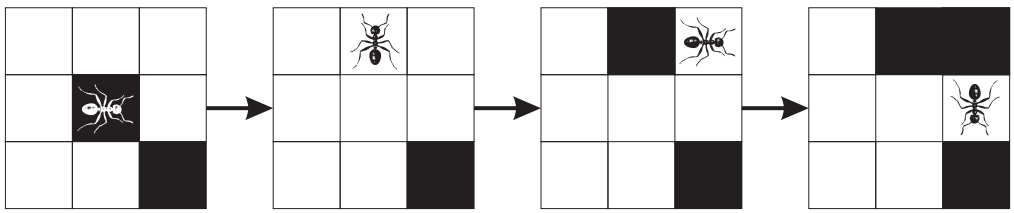
   (1) Change the orientation of the ant 90° to the left or right, depending on whether the color of the current cell is Black or White, respectively, and then swap the color

   $$\text{White} \leftrightarrow \text{Black}$$

   of the cell,

   (2) Move the ant one cell to the direction of its new orientation.

   For example, here are three consecutive moves of an ant:



   Prove that starting from any initial coloring of the two-dimensional tape, the ant will visit infinitely many different cells.